The Future of Functional ECO Automation and Logical Equivalence Checking for Advanced Digital Design Flows

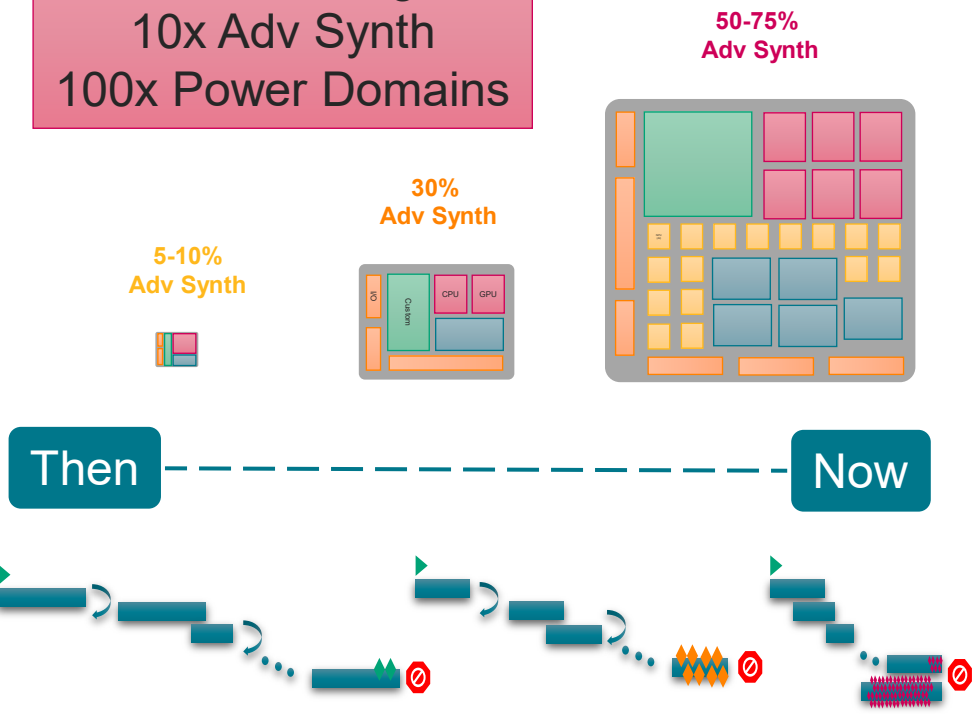
Dr. Zhuo Li, Sr. Group Director, Cadence Design Systems
David Stratman, Product Management Director, Cadence Design Systems

cadence®

Design Trends Create Formal Verification Challenges

Logical Equivalence Checking & ECO

100x SoC Logic
10x Adv Synth
100x Power Domains



100x More ECOs
1000x Larger ECOs
Complex ECO Process
+ All the LEC Challenges

LEC became **too hard**…
Designers **forfeit PPA** to get EQ

+

Automated ECOs are now
Strategic parts of the SoC **Plan**...
But remain **difficult to deploy** at scale

What is ECO (Engineering Change Order)

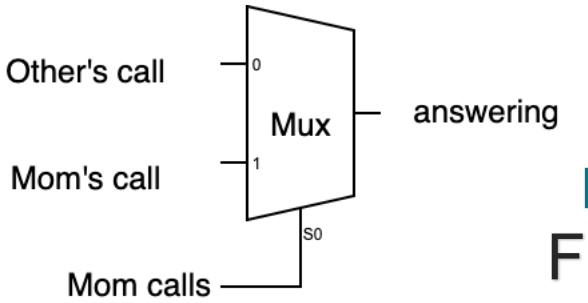
- Everyone always do last-minute changes, so as chip designer.
- For example: my answering Protocol 25 years ago

```
If (Mom calls)
        get Mom's call
else
        get Other's call
```

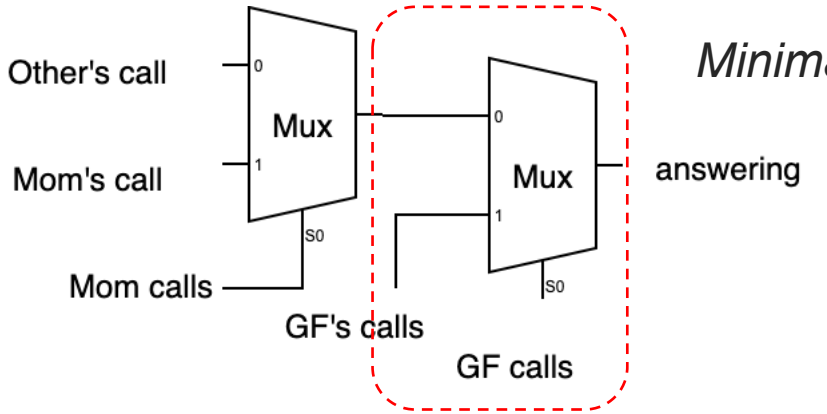
Protocol change

```
If (GF calls)
        get GF's call
else if (Mom's call)
        get Mom's call
else
        get Other's call
```

Implementation
(Months effort $$$$)



Functional ECO



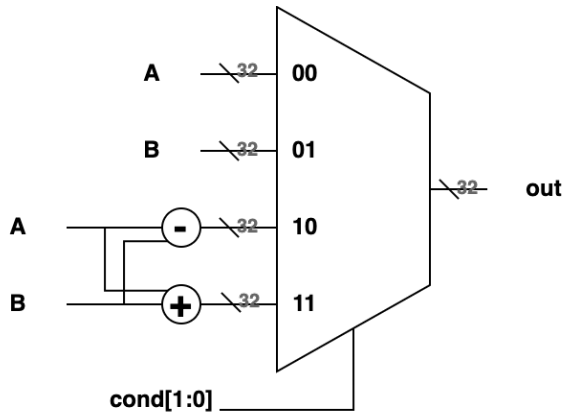
Minimal Patch

Based on existing implementation, generate the minimal patch ($) such that we can match the new spec(new protocol, design)

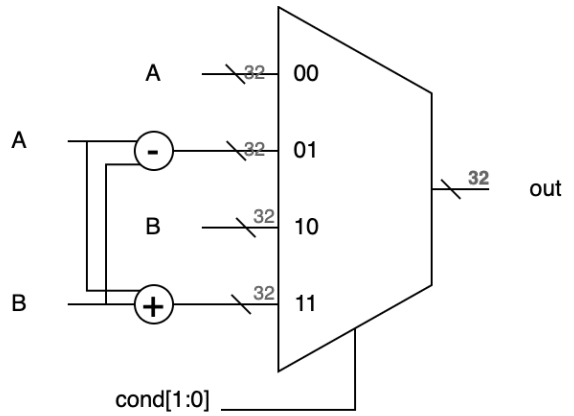
How?

ECO Example

Original implementation

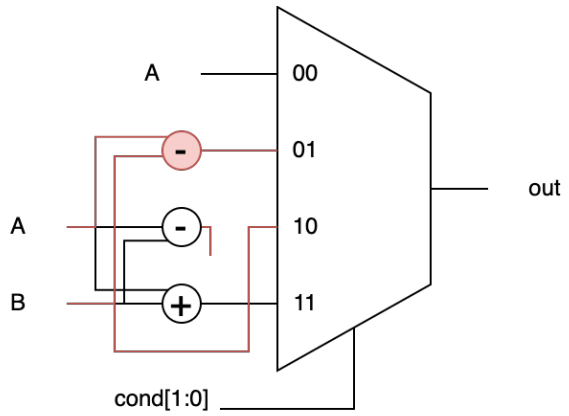


Desired function



EQ

Change



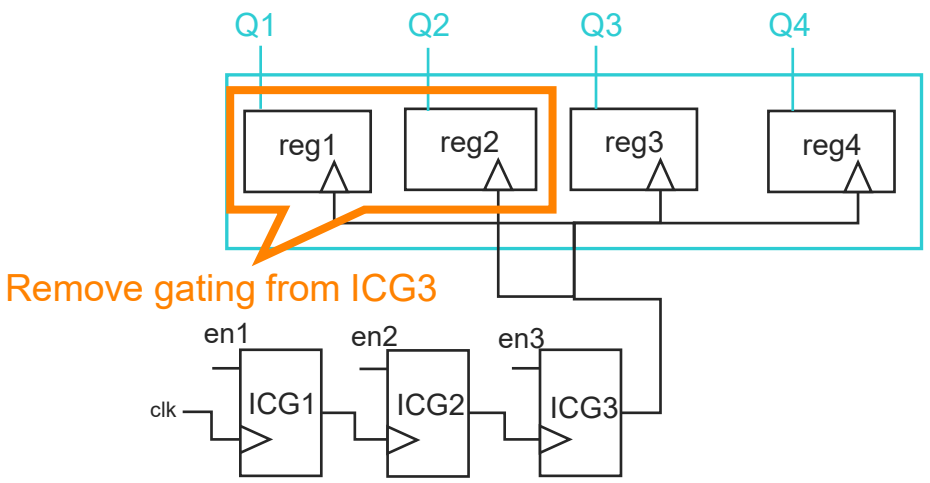
Patch (Cost $$$):

- add “-” subtractor
- Disconnect the original subtractor
- change hundreds of wires

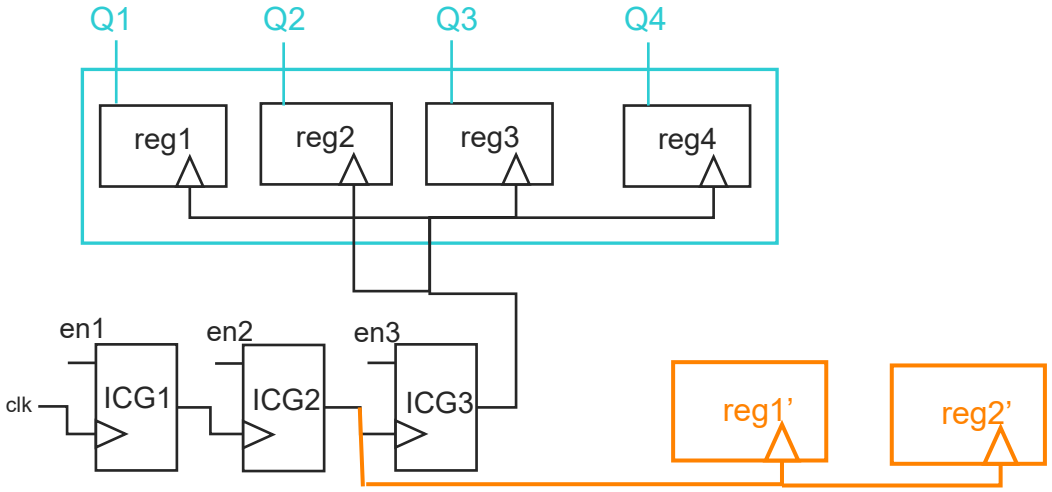
Optimal solution: 2 wires only

Which ECO Solution is the Best?

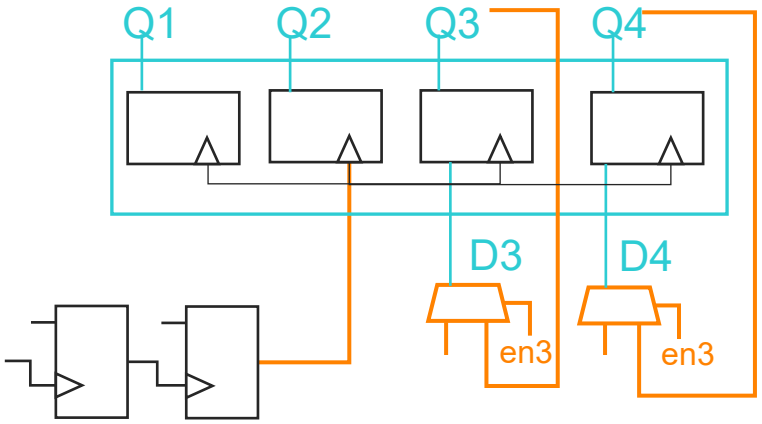
- ECO change: Remove gating from ICG3 on reg1 and reg2



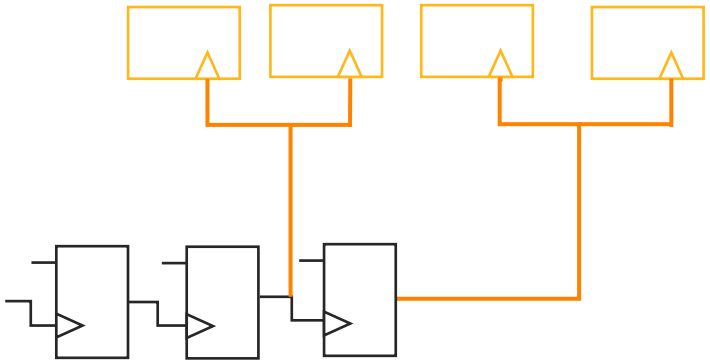
G1



Solution 1



Solution 2



Solution 3

AI/ML Could Help Identify Optimal Functional ECOs

Explore the Full Search Space

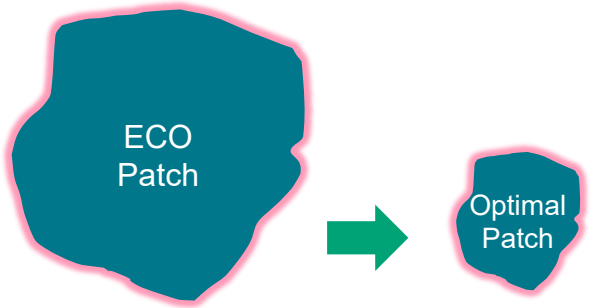
- 20+ different ECO recipe and tool options → $>2^{20}$
- Rapidly explore full ECO patch solution space

Leverage Historical Project Data

- Dynamic categorization of ECO'd RTL
- Analyze RTL to drive ECO strategy priority

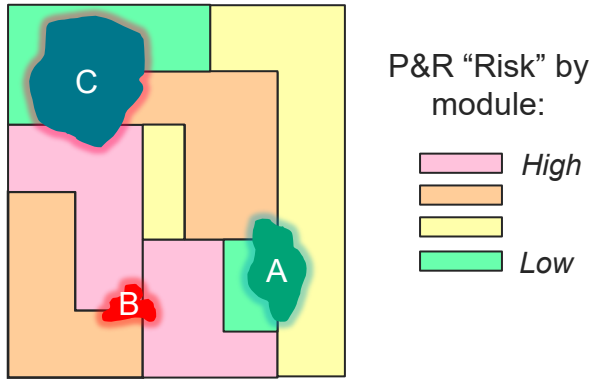
Analyze Downstream ECO Impact

- Explore ECO patch PPA / congestion / DRC-LVS, etc.
- Measure final ECO results



Smaller ECOs

Faster Time to ECO

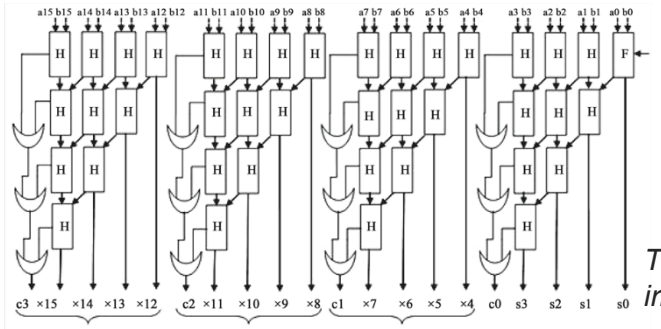


Identify the Best Overall ECO

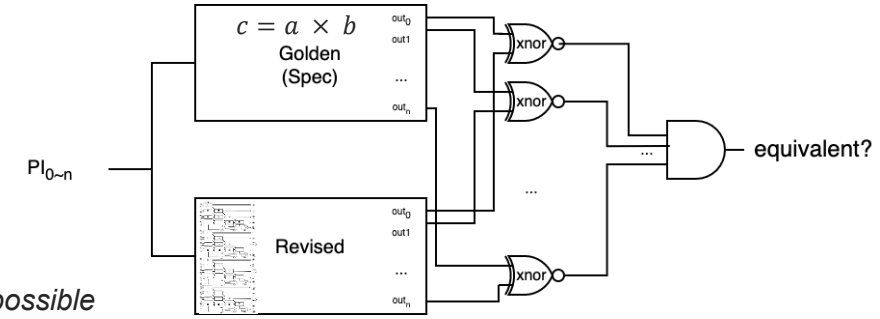
Formal Verification: Modeling the problem in formal ways

- Are they equivalent?

$$c = a \times b$$

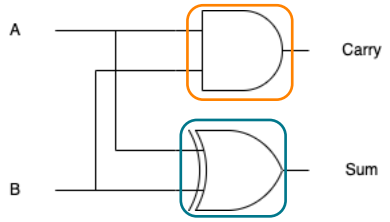


Compare model



There are countless possible implementations for a multiplier

- Simulation based is not applicable, where complexity is $O(2^n)$ to be complete where *n* is number of input variables. Random simulation (semi-formal) cannot cover all possible space.
- Formal verification is complete and can be very fast, but how? Modeling to be solved efficiently
  - Ex: model the Half adder as the SAT instance as



formal

A	B	Sum (S)	Carry (C)
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Logic circuit 101

formal

$(A \vee \neg C)\ (B \vee \neg C)$
$(\neg A \vee \neg B \vee C)$
$(A \vee B \vee \neg S)$
$(A \vee \neg B \vee S)$
$(\neg A \vee B \vee S)$
$(\neg A \vee \neg B \vee \neg S)$

7 clauses

Algorithm 101

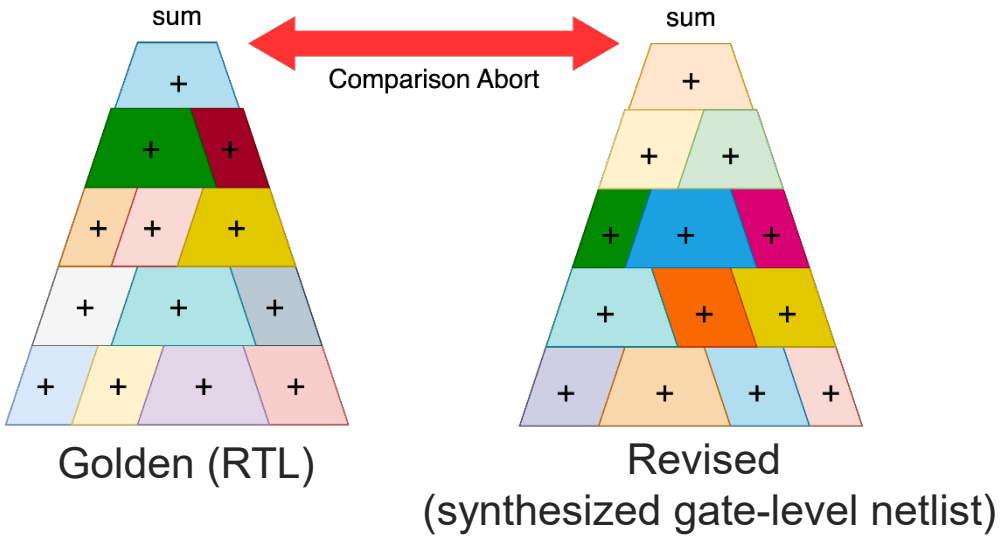
- SAT solver can solve the 10^4~10^6 clauses very quick, but still an NP complete problem.

To handle million/trillions Boolean logics, we need more learnings/heuristics, and AI.

LEC Datapath Challenges

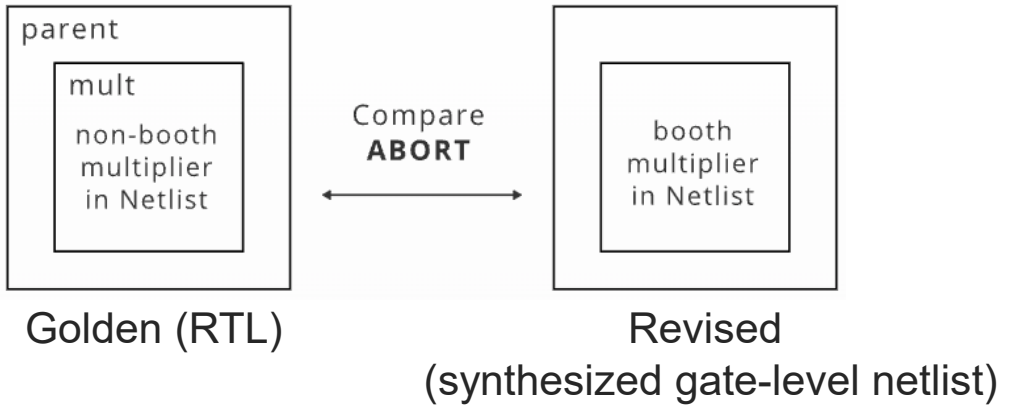
- High implementation complexity in datapath
  - Example: complexity of 32bits * 32bits multiplier
    - 32 bits * 32 bits = 32bit + 32bit + … + 32bit

31 adders



  - Complexity of different addition implementation is 32!
    (even ignoring how partial product is generated)

- Different multiplier architecture
  - Wallace tree
  - Booth encoding (radix-2, radix-4, … etc)
  - Partial products



Ask for Research Community

- Sequential ECO: Can we generate a patch such that the netlist becomes sequential equivalence.
- Full Flow ECO
- Generic Boolean solver for the datapath problem. (A Boolean representation such that the solver can performs learning, rather than using the CNF(SAT) form)
- Equivalence Checking for the “non-exact function” like the floating multiplier or faithful rounding.
- Functional Equivalence with High-level synthesis (Not cycle accurate EC, but the EC after multiple cycles)
- Redundancy Equivalence Checking. Checking the equivalences but also guarantee the redundancy are the same. (for safety application)

Thank You!

- Acknowledgements
  - Jacky Hsu, Kwangsoo Han, and whole Conformal Team

- Contact
  - Openings in Conformal (LEC/ECO), Modus (DFT), Stratus (High Level Synthesis)